REMARKS

By this amendment, Claims 1, 6, 7, 10, 12, 15, 16, 18, 39-42, 44-51, and 53-56 are amended, and no claims are added or cancelled. Hence, Claims 1-4, 6-13, 15-20, 39-42, 44-51, and 53-58 are pending in the application.

The amendments to the claims as indicated herein do not add any new matter to this application. Furthermore, amendments made to the claims as indicated herein have been made to exclusively improve readability and clarity of the claims and not for the purpose of overcoming alleged prior art.

Each issue raised in the Final Office Action mailed July 16, 2007 is addressed hereafter.

I. SUMMARY OF THE REJECTIONS

Claims 1-4, 6-13, and 15-17 stand rejected under 35 U.S.C. § 101 as allegedly directed to non-statutory subject matter. This rejection is respectfully traversed.

Claims 1-4, 7-11, 13, 16-20, 39-42, 45-49, 51, and 54-58 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over U.S. Patent No. 6,950,822 issued to Idicula et al. ("*Idicula*") in view of U.S. Patent No. 5,911,143 issued to Deinhart et al. ("*Deinhart*"). This rejection is respectfully traversed.

Claims 6, 15, 44, and 53 stand rejected under 35 U.S.C. 103(a) as allegedly being unpatentable over *Idicula* in view of *Deinhart*, and further in view of U.S. Patent No. 6,233,576 issued to Lewis ("*Lewis*"). This rejection is respectfully traversed.

Claims 12 and 50 stand rejected under 35 U.S.C. 103(a) as allegedly being unpatentable over *Idicula* in view of *Deinhart*, and further in view of Official Notice. This rejection is respectfully traversed.

II. ISSUES NOT RELATING TO THE CITED ART

Claims 1-4, 6-13, and 15-17 stand rejected under 35 U.S.C. § 101 as allegedly directed to non-statutory subject matter. In the "Response to Arguments," the Final Office Action asserted that "the methods are not embodied on a form of computer-readable medium, said methods are not functional as they may not be performed by a computer." Neither the U.S. Constitution, the United States Code (particularly 35 U.S.C. § 101), nor the Code of Federal Regulations requires a process to be embodied on a form of computer-readable medium. The claim does not recited printed matter. A process is one of the explicitly-enumerated statutory classes under 35 U.S.C. § 101. If it is the Final Office Action's position that the steps recited in Claims 1 and 10 may be mental steps, then it is also respectfully noted that the claims recite steps that cannot be mentally performed. For example, Claim 1 explicitly recites "creating and storing in a filesystem of an Operating System a file that represents the resource" (emphasis added).

However, to expedite resolution of this case, Claims 1 and 10 are amended to recite a "computer-implemented method." Therefore, the method is implemented by a computer. Furthermore, Claims 6, 7, 15, 16, 44, 45, 50, 53, and 54 are amended to replace "only if" with "when" as suggested in the Final Office Action (page 9).

III. ISSUES RELATING TO THE CITED ART

Claims 1-4, 7-11, 13, 16-20, 39-42, 45-49, 51, and 54-58 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over *Idicula* in view of *Deinhart*.

A. CLAIM 1

Claim 1 recites:

> A computer-implemented method for controlling access to a resource, the method comprising the steps of:
> **creating and storing in a filesystem of an Operating System a file that represents the resource;**

receiving user-identifying information from a user requesting access to the resource, wherein the user-identifying information comprises a role associated with the user, wherein the role is determined from a user identifier uniquely associated with the user and from a group identifier associated with a group that includes the user;

receiving a resource identifier associated with the resource;

creating an access identifier based on the user-identifying information and the resource identifier, wherein the access identifier is formatted as a file attribute that is used by the Operating System to manage file access;

calling the Operating System to perform a file operation on the file by providing the access identifier to the Operating System to attempt access to the file; and

granting the user access to the resource only when the Operating System call successfully performs the file operation;

wherein the file operation on the file representing the resource is selected from a group consisting of opening the file, closing the file, deleting the file, reading from the file, writing to the file, executing the file, appending to the file, reading a file attribute, and writing a file attribute. (emphasis added)

In an embodiment, resources may be a software application, software application message, server, router, switch, file, directory, etc. (Specification, paragraphs 27 and 28). A file is created and stored in a filesystem of an Operating System. The file represents a particular resource. Subsequently, user-identifying information is received from a user who is requesting access to that resource. A resource ID associated with the resource is also received, e.g., from the user. Subsequently, an access ID is created based on the user-identifying information and the resource ID. The access ID is formatted as a file attribute that is used by the Operating System to gain access to the file. The Operating System is then called to perform a file operation (e.g., open, write, delete) on the file that represents the resource. This is done by providing the access ID to the Operating System to attempt access to the file. The user is granted access to the particular resource when the Operating System call successfully performs the file operation.

Because Operating Systems calls are used to attempt access to a file, a complex security mechanism is avoided and application software developers are not required to learn a specific API and to write code against it (see specification, paragraph 3). Claim 1 uses the filesystem of

an Operating System (OS) to manage access to files. The OS filesystem is one of the most well-known and easy to write code systems in a computer system (see specification, paragraphs 4, 7, and 8).

In contrast, *Idicula* does not teach anything substantial about (1) an Operating System, (2) a filesystem, or (3) a file that represents the resource. Instead, *Idicula* teaches a technique for servicing requests for database services (Abstract). Specifically, *Idicula* teaches that rather than deleting session objects associated with a particular session when the corresponding connection ends, some of those session objects are saved and reused with new connections (col. 6, lines 6-9). Thus, the resources that would be consumed to delete and recreate session objects when connections are ended and started, respectively, are conserved (col. 6, lines 11-15).

Due to these significant differences, *Idicula* fails to teach or suggest numerous features of Claim 1, which features were discussed and distinguished in the reply filed on May 2, 2007.

Further, Claim 1 recites: "creating and storing in a filesystem of an Operating System a file that represents the resource." The Final Office Action equates the database session object of *Idicula* with a "file" as recited in Claim 1 (page 10). The Final Office Action then asserts that a "session object file would be stored in some sort of a filesystem" (page 10). This is incorrect. Files and filesystems are well-known in the art. Even when a copy of a file is read into volatile memory, the file is stored in non-volatile memory, such as a floppy drive or a hard drive. The same cannot be said about database session objects, which typically only reside in volatile memory and do not relate to files. A filesystem is a system, used by an operating system, to organize and keep track of directories and files. By equating a session object to a file and stating that a session object would be stored in "some sort of a filesystem," the Final Office Action unreasonably broadens the meaning of the terms "file" and "filesystem" and fundamentally alters the well-founded meaning of those terms.

Additionally, *Idicula* specifically refers to files and a filesystem and distinguishes files from database session objects. A portion of col. 2, lines 6-18 of *Idicula* states: "to support database requests for **hierarchical data, such as files and folders of a file system** stored in a repository within the database, the **database session object is extended to include information about a root container for the hierarchy**" (emphasis added). Col. 5, lines 30-31 of *Idicula* further teach that files (and folders) are stored in a database. Therefore, according to *Idicula*, a database session object is different than a file. Also, based on the interpretation of the Final Office Action, a database session object would have to be stored in the filesystem referenced in col. 2, line 10 of *Idicula*. However, *Idicula* fails to teach or suggest that a database session object is stored in the referenced filesystem.

Claim 1 recites that "a file represents the resource." The Final Office Action equates the database of *Idicula* with the "resource" of Claim 1. Thus, it is the position of the Final Office Action that a database session object represents a database. This position is both incorrect on its face and does not make sense.

Claim 1 further recites "calling the Operating System to perform a file operation on the file by providing the access identifier to the Operating System to attempt access to the file." In response to Applicants' argument that *Idicula* fails to teach or suggest this feature of Claim 1, the Final Office Action states:

> Idicula discloses a system wherein the contents of a session object (e.g., session information) are checked to see if a session has already been created for a client. Upon finding that a session object associated with the client is indicated in the process state object (i.e., the file operation), the client is then allowed to received the requested database services from the database (page11).

No where does *Idicula* teach or suggest that an **operating system is called**. *Idicula* specifically refers to an operating system at col. 1, lines 58-61. Even if the finding of a database session

object could be considered a "file operation", *Idicula* fails to teach or suggest that the referenced operating system is called to perform the "file operation" on the session object.

Claim 1 further recites "granting the user access to the resource only when the Operating System call successfully performs the file operation" (emphasis added). The Final Office Action, as indicated previously, equates the database of *Idicula* with the recited "resource" of Claim 1. The Final Office Action asserts that:

> a client seeking database services passes a request to an Operating System. Within the passing of the request, client information associated with the session and resource are called and passed on to the Operating System. With the client information, the Operating System attempts to read the session object, searching for the presence of an existing session for the client. Wherein the Operating System successfully finds an existing session for the client, the client is then granted access to the database" (page 11).

It is respectfully submitted that this is incorrect. As indicated previously, *Idicula* only mentions "operating system" once, at col. 1, line 59. No where does *Idicula* teach that: (1) a client passes a request to an Operating System; (2) client information associated with a session is passed to an Operating System; or (3) an Operating System attempts to read a session object by searching for the presence of an existing session, as the Final Office Action alleges. Not only does *Idicula* distinguish between an Operating System and a database server, *Idicula* explicitly teaches that a database server performs the above steps alleged in the Final Office Action. For example, col. 7, lines 21-23 of *Idicula* states, "The **database server** determines whether a session has already been created for this client by checking the contents of process state object 130" (emphasis added). As another example, col. 4, lines 15-18 of *Idicula* states, "The techniques described hereafter allow a **database server** to more efficiently service more requests for database services" (emphasis added). As yet another example, col. 4, lines 42-44 of *Idicula* states, "The database server 110 includes memory 120 on the database server host computer, which is allocated for use by the **database server** 110. The database server 110 maintains several data

structures in memory 120" (emphasis added), which data structures include the database session objects and process state objects.

The Final Office Action further equates: (1) the reading of a database session object with the recited "file operation" of Claim 1; and (2) the finding of an existing session for a client with the recited step of "granting the user access" of Claim 1. Even if these analogies could be made, *Idicula* would still fail to teach or suggest that a user is granted access to a database only when an existing database session object for a client is read and found. *Idicula* teaches that if a database server does not find any available session objects for a client, then access to the database is **still granted** (col. 7, lines 36-43, 51, and 57-58; see also FIG. 2, steps 230-240). The database server simply creates a new database session object and associates that session object with the client.

Based on the foregoing, *Idicula* fails to teach or suggest numerous features of Claim 1. Additionally, *Dienhart* is not used to show the features of Claim 1 that are not taught or suggested by *Idicula*. Therefore, Claim 1 is patentable over *Idicula* and *Deinhart*. Reconsideration and withdrawal of the rejection of Claim 1 under 35 U.S.C. § 103(a) is therefore respectfully requested.

B. CLAIMS 10, 18, 39, 48, AND 56

The Office Action stated the same reasons in rejecting Claims 10 and 18 to those in rejecting present Claim 1. Also, Claims 39, 48 and 56 recite features discussed above that make Claim 1 patentable over *Idicula* and *Deinhart*. Therefore, for at least the same reasons set forth above by the Applicant in connection with present Claim 1, it is respectfully submitted that each of Claims 10, 18, 39, 48 and 56 is patentable over *Idicula* and *Deinhart*.

C. DEPENDENT CLAIMS

The dependent claims not discussed thus far are dependent claims, each of which depends (directly or indirectly) on one of the independent claims discussed above. Each of the dependent claims is therefore allowable for the reasons given above for the claim on which it depends. In addition, each of the dependent claims introduces one or more additional limitations that independently render it patentable. However, due to the fundamental differences already identified, to expedite the positive resolution of this case, a separate discussion of those limitations is not included at this time. The Applicant reserves the right to further point out the differences between the cited art and the novel features recited in the dependent claims.

IV. CONCLUSIONS & MISCELLANEOUS

For the reasons set forth above, all of the pending claims are now in condition for allowance. The Examiner is respectfully requested to contact the undersigned by telephone relating to any issue that would advance examination of the present application.

A petition for extension of time, to the extent necessary to make this reply timely filed, is hereby made. If applicable, a law firm check for the petition for extension of time fee is enclosed herewith. If any applicable fee is missing or insufficient, throughout the pendency of this application, the Commissioner is hereby authorized to any applicable fees and to credit any overpayments to our Deposit Account No. 50-1302.

Respectfully submitted,
HICKMAN PALERMO TRUONG & BECKER LLP

Dated: October 15, 2007

/DanielDLedesma#57181/
Daniel D. Ledesma, Reg. No. 57,181

2055 Gateway Place Suite 550
San Jose, California 95110-1093
Telephone No.: (408) 414-1229
Facsimile No.: (408) 414-1076